KING FAHD UNIVERSITY OF PETROLEUM & MINERALS
Accounting & MIS Department
MIS105- Introduction to Computer Applications
Instructor: Irfan Ahmed Ilyas

1. Use of Visual Basic for Applications (VBA) code

Some *important* information about **VBA**:

- **VBA** stands for Visual Basic for Application.
- It is an ***event driven*** programming language (different objects can respond to different events like mouse click, double click, etc.)
- **VBA** code is developed in units called procedures or module.
- These event procedures (a set of **VBA** commands) are attached to some object's event.
- A **VBA** procedure starts with the reserved word **Sub** and ends with **End Sub**.
- **VBA** is used for more or less the same purpose as macros are having. However a VBA code allows more control over the application behavior then a macro does, in terms of.
    1. Data validation (e.g. password checking etc.)
    2. Dynamic alteration of Object Properties
    3. Facilitating Data entry
    4. Enhanced Communication with User

Where to start in VBA coding?

Whenever you are asked to write a VBA code for some situation, you are supposed to find out the following two answers.

Question#1. Which object (button, list box, etc) is supposed to provide the needed functionality

Question#2. Which specific event of the object (identified in Question#1) the code (VBA procedure) should be linked [for example, a button object a "mouse click" is mostly used].

For example, In a situation where you are asked to provide a button object which can open a form object when somebody clicked over it. The answers for the above two questions are:

Answer#1. The button object is the one where a VBA program is needed.

Answer#2. The button object's event, named OnClick, will be used to attach the code to the button.

Rule for writing an Event Procedure.

Whenever writing some VBA code, always make sure that the code appear in between the following two code lines.

```
Option Compare Database

Private Sub Text0_Click()

End Sub
```

New code must be added here

Basic Activities using VBA Code

1. Using MS Access objects

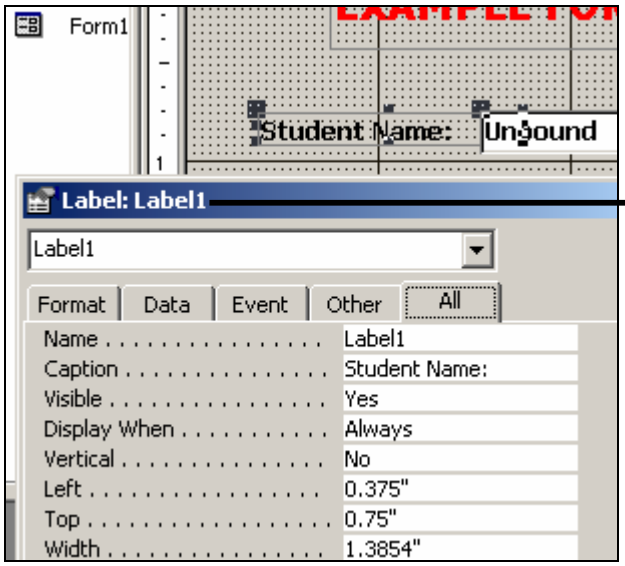
1. In MS Access every thing like table, query, form, label, textbox, combo box etc. is treated as an Object.
2. Every object has a set of properties which can be altered through the object property dialogue.
3. Every OBJECT has a unique name assigned automatically by MS Access. Can be changed by the user.

Hands On 1
Open the given form, named Form1. Identify the objects on the form and note down their names.

Steps:

- Open Form1 in the design view.
- You can identify the following objects.
    1. Two Labels, named Label1 & Label2.
    2. One text box, named Text0.

Note: Name of the objects can be determined in the window title of object's property dialogue box.



Name of the object (currently selected)

2.Using a MS Access Object: Reading and changing object's property value

The property value for different object's property can be changed in design view as well as during run time (using VBA code).

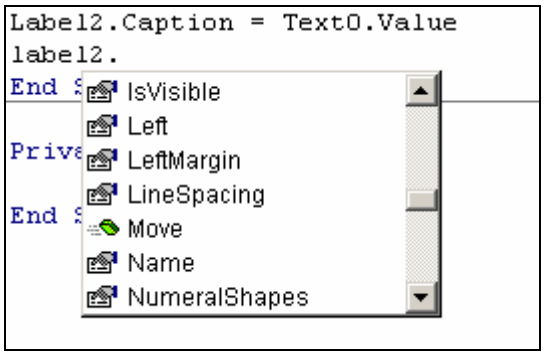
Hands On 2
For Form1, write a small code (on a command button 'On Click' event) which will copy the title in the text box into the Form's main label (Label2).

Steps:

1. Paste a new command button and write the Event Procedure in the On Click event.
2. Make sure you already noted down the object names which you need in your VBA code lines. For example, you need to read the text box value and copy it in as the label's text.
    - Name of the text box (in the example): Text0
    - Name of the label (in the example): Label2
3. Make sure you know the names of the properties which you are dealing with. For example, to read the contents of a textbox and copy it as the label's text the following properties can be used.
    - **Text box property: Text0.Value**
    - **Label property: Label2.caption**
4. Write the following code
    - **Label2.caption = Text0.Value**
5. Run the form and test your code.

Notes.

- When you write the name of the object and press a dot, a small window will automatically be opened showing a list of object's properties. In case you are not experience such a window, it means the name spelling for the object is having some problem.
- Most of the property names are appearing with a hand icon (shows it is a property). However some of them appear with a greenish icon (shows it is a method- a small hidden program which when runs to do some operation on the object).



3.Using a MS Access Object: Calling a object's method

By using the same syntax for accessing an object's property, an object's method can also be called.

Hands On 3
For Form1, write a small code (on a command button 'On Click' event) which will call the method named 'Move' on Label2 object.

Steps:
In the VBA code of the command button, call the method move for Label2 using the following statement.

Label2.Move 10, 10

Note: The effect of move method is to change the position of the label object within its window (form). The method requires the caller to specify the position coordinates (X,Y) in front of the call. The method can also change the width and height of the label if 2 more numbers are provided.

4.Using DoCmd Object: Calling a object's method
Docmd is a special object in MS Access which doesn't have any property defined. Only methods can be called.
Docmd object is not a visual object like command buttons, labels or text boxes etc. Instead it is an internal object which is always available for calling different methods.
The use of Docmd object is to call different MS Access operations (available in MACRO list) directly in the VBA code.

Hands On 4
For Form1, write a small code (on a command button 'On Click' event) which will make use of docmd to call different macro like actions including BEEP, CLOSE etc.

Steps:
Write the following code on 'OnClick' event of a command button on Form1.

```
Docmd.Beep
Docmd.OpenForm "MainForm"
Docmd.Close "Form1"
```

3. Using Basic VBA Statement

a. Assignment Statements

These statements let the programmer

- To assign a value to an object's property OR

- To assign a value to a data variable (similar to variables in mathematical formulas)

The values used in the assignment may be of numeric or alphabetical in nature (depending upon the type of property or variable to whom the value is being assigned to).

Hands On 5

In Form1, write a VBA code (on a command button 'On Click' event) use the following assignment statements.

STEPS:

1. Assigning an alphabetical (character string) value to a label's caption property (character type property)

```
Label2.caption = "This a new heading."
```

2. Assigning a numeric value to label's FONTSIZE property (numeric type property)

```
Label2.fontsize = 20
```

3. Defining two variables, named 'StudentName' and 'Score' of the type 'string' and 'Integer' respectively and then assigning some values to these variables.

```
'Variable definitions
Dim  StudentName  as String
Dim Score as Integer

'Variable assignments
StudentName = "Mohammad"
Score = 80
```

b. Input/ output statements

OUTPUT STATEMENT: MSGBOX

In VBA, the most common statement used for output is MsgBox. The statement will produce a small dialogue box (an OK button only) with the given message printed inside the dialogue. The syntax is as follows:

Msgbox <value or variable to be printed inside the message box> [, vbOK | vbExclamation …]

INPUT STATEMENT: MSGBOX

For input, VBA uses most commonly InputBox statement. The syntax is as follows:

<Variable to save the input value> = InputBox ("Message for the user", "Dialogue Title")

Hands On 6

In Form1, write another VBA code (on another command button's 'On Click' event). The code should ask your name in a dialogue box titled 'Name Input' and then saved it in a variable named YourName. The code then prints the value of YourName into a

MessageBox.

Steps:
Enter the following code into OnClick event of another command button.

```
Dim YourName As String
YourName = InputBox ("What is your last name?")
Msgbox "Your Name is: " & YourName
```

ALTERNATIVE APPROACH FOR READING INPUT DATA: TEXTBOX OBJECT

Hands On 7

In Form2, write VBA code on the command button's 'On Click' event.

The code should ask your name in a textbox box (already exist on the form) labeled 'Your Name Here. When the user presses OK button, the code should save the value in a variable named YourName. The code then prints the value of YourName into a MessageBox.

Steps:
Enter the following code into OnClick event of another command button.

```
Dim YourName As String
YourName = Text19.Value
Msgbox "Your Name is: " & YourName
```

c. Decision statements

IF-THEN-ELSE: Simple Decisions

In VBA, the statement used to take a conditional action is IF-THEN-ELSE statement.

Syntax:

```
IF <condition to test> THEN
        <Statements to be done in case of a true condition>
ELSE
        <Statements to be done in case of a false condition>
ENDIF
```

Hands On 8

In Form3, write VBA code on the command button's 'On Click' event.

The code should read the average temperature value in the textbox box (already exist on the form) labeled 'Average Temperature Here'. When the user presses OK button, the code should save the value in a variable named AvgTemp (type Single).

The code then prints the message 'Sunny day' in a message box if the value read is more than 25, otherwise (if value is not more than 25) the code prints

'Normal day'.

Steps:
Enter the following code into OnClick event of another command button.

```
Dim Avgtemp As Single
Avgtemp = Text20.Value

If (Avgtemp > 25) Then
   MsgBox "Sunny Day"
Else
   MsgBox "Normal Day"
End If
```

CASE Statement: Complex Decisions
In VBA, the statement used to take a conditional action is CASE statement.

Syntax:
```
SELECT CASE  <variable to test>
        CASE <condition#1>:
                Statements

        CASE <condition#2>:
                Statements

        CASE ELSE
                Statements
END SELECT
```

Hands On 9

In Form3, write VBA code on the command button's 'On Click' event.

The code should read the average temperature value in the textbox box (already exist on the form) labeled 'Average Temperature Here'. When the user presses OK button, the code should save the value in a variable named AvgTemp (type Single).

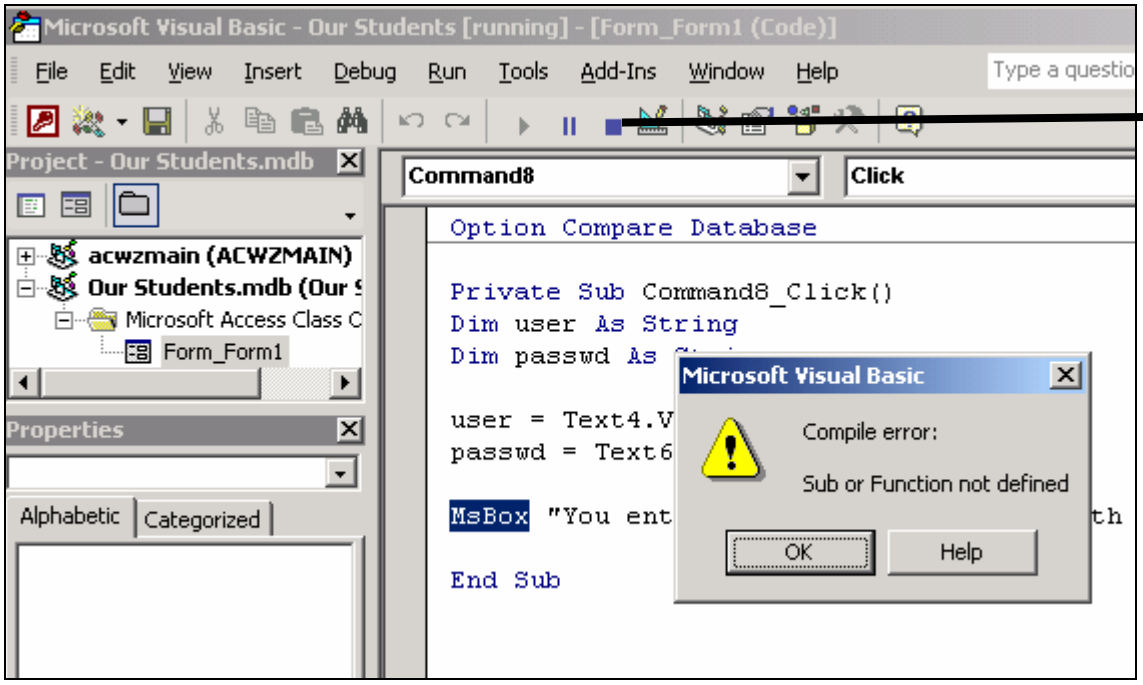
The code then prints the following messages:
AvgTemp > 40: 'Hot Day'
AvgTemp > 25: 'Sunny Day'
AvgTemp > 15: 'Normal Day'
AvgTemp > 5: 'Cold Day'

```
Dim Avgtemp As Single
Avgtemp = Text0.Value

Select Case (Avgtemp)
```

```
    Case Is > 40:
       MsgBox "Hot day"
    Case Is > 25:
       MsgBox "Sunny day"
    Case Is > 15:
       MsgBox "Normal day"

    Case Else
       MsgBox "cold day"
End Select
```

2. Correcting coding errors in VBA (Debugging code)

In case a mistake was done in the code (like misspelling for some command), an error window will appear as follows:



Things to do in case of an error.

1. Read the error carefully.
2. Push the **Reset button** to stop the debugger.
3. Fix the error.
4. Close the debug window and run the form again.